$G_5$ ID Number Allocation Procedure

Next, the procedure for allocating an ID number to each decoder is described below with reference to Figs. 10 and 11. First, a program starts at step (イ), and at step (ロ), a decoder (3A) checks if a data sequence for the ID number allocation as illustrated in Fig. 10 is received from a controller (1). At step (ハ), the decoder (3A) determines whether the information sent from the controller (1) is the data sequence for the ID number allocation thereto, and if not, the program goes to step (ヘ) and exits from this routine. If the information is the data sequence for the ID number allocation, the decoder (3A ) stores the ID number included in the data sequence as its own ID number, and initialization is carried out.

Next, the decoder (3A) increments its own ID number by one at step (ホ), and outputs the value to an AUX port as the ID number of a decoder (3B) at the next stage, and the program exits from the routine at step (ヘ).

In the same manner, the decoder (3B) stores the ID number given by the decoder (3A) as its own ID number, and initialization is carried out. The decoder (3B) increments its own ID number by one, and outputs the value to the AUX port as the ID number of a decoder (3C) at the next stage. The same routine is sequentially repeated for decoders (3D) through (3I), and an ID number is finally allocated to all of the decoders (3A) through (3I).

C．ID番号の割付け

次にデコーダにID番号を割付ける手順を第10図及び第11図を参照して説明する。先ず、ステップ（イ）でプログラム開始し、ステップ（ロ）でデコーダ（3A）はコントローラ(1)より第10図に示すようなID割り付けのデータシーケンスが送られているかをチェックする。ステップ（ハ）でデコーダ（3A）はコントローラ(1)より送出されてくる情報がID割り付けデータシーケンスか否かを判断し、そうでなければステップ（ヘ）に進んでプログラムを終了し、そうであれば当該データシーケンスに含まれるID番号を自己のID番号として記憶保存する。そして初期設定される。

次にデコーダ（3A）はステップ（ホ）で自己のID番号を1つインクリメントして次段のデコーダ（3B）のID番号としてAUXポートに出力し、ステップ（ヘ）にてプログラムを終了する。

同様にデコーダ（3B）はデコーダ（3A）より供給されたID番号を自己のID番号として記憶保存し、初期設定される。そしてデコーダ（3B）は自己のID番号を1つインクリメントして次段のデコーダ（3C）のID番号としてAUXポートに出力する。以下（3D）～（3I）に付いても同様の動作が順次行われ、全てのデコーダ（3A）～（3I）に対するID番号の割り付けが終了する。

C．外部同期

次に各デコーダに外部同期をかける場合、つまりコントローラ(1)からの同期制御信号によりデコーダ（3A）～（3I）を一斉に駆動させる場合を第12図及び第13図を参照して説明する。第12図はコントローラ(1)の動作で、第13図はデコーダ（3A）～（3I）の動作である。先ず、ステップ（イ）でプログラム開始し、ステップ（ロ）でコントローラ(1)はI／Oインターフェース（15）から出力される同期制御信号を一方のレベル例えばローレベルとする。次にステップ（ハ）でコントローラ(1)はデコーダ（3A）～（3I）に対して全てのデータを送る。ステップ（ニ）でコントローラ(1)は全てのデータ送信完了後にI／Oインターフェース（15）から出力される同期制御信号を他方のレベル例えばハイレベルにする。ステップ（ホ）でプログラムを終了する。

一方、デコーダ（3A）～（3I）は各々ステップ（イ）でプログラム開始し、ステップ（ロ）でCOMポートよりデータを受信する。ステップ（ハ）で受信データをAUXポートに出力する。ステップ（ニ）でコントローラ(1)のI／Oインターフェース（15）よりデコーダのI／Oインターフェース（25）に供給されている同期制御信号がハイレベルか否かを判断し、ハイレベルでなければすなわちローレベルであればステップ（ロ）へ戻り、ハイレベルであればステップ（ホ）に進んでデータをデコード開始する。ステップ（ヘ）で、データ終了か否かを判断し、データ終了でなければステップ（ニ）へ戻り、データ終了であればステップ（ト）に進んでプログラムを終了する。

つまり、デコーダ（3A）～（3I）はコントローラ(1)からの同期制御信号がローレベルの間はデータを取り込むだけでデコードは行われず、同期制御信号がハイレベルになると一斉にデコード開始する。

C．フローコントロール

次に直列接続されたデコーダのデータのオーバフローが検出されたら、前段のデコーダに対してデータ出力の停止を命令するフローコントロールの手順を第14図及び第15図を参照して説明する。先ず、第14図に於いてコントローラ(1)はCOMポート及びAUXポートに対してワークRAM（12）上に夫々送信バッファTC及び受信バッファRCと送信バッファTA及び受信バッファRAを有しており、ここではAUXポートの送信バッファTA及び受信バッファRAのみを示している。また、各デコーダもCOMポート及びAUXポートに対してワークRAM（12）上に夫々送信バッファTC及び受信バッファRCと送信バッファTA及び受信バッファRAを有している。そしてコントローラ(1)のAUXポートの送信バッファTAのデータはデコーダ（3A）のCOMポートの受信